昵称：Bodi
园龄：8年4个月
粉丝：71
关注：5
+加关注

Zookeeper 3、Zookeeper工作原理（详细）

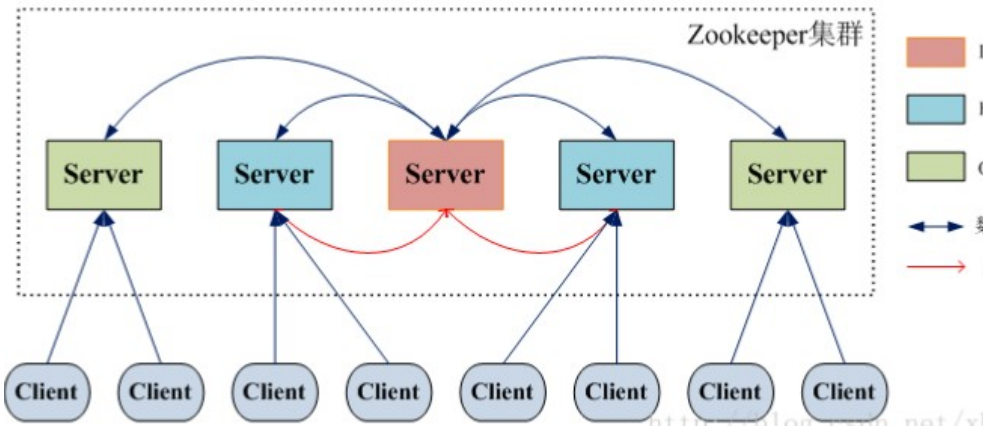
1、Zookeeper的角色

» 领导者（leader），负责进行投票的发起和决议，更新系统状态

» 学习者（learner），包括跟随者（follower）和观察者（observer），follower用于接受客户端请求并想客户端返回结果，在选主过程中参与投票

» Observer可以接受客户端连接，将写请求转发给leader，但observer不参加投票过程，只同步leader的状态，observer的目的是为了扩展系统，提高读取速度

» 客户端（client），请求发起方



角色		描述
领导者（Leader）		领导者负责进行投票的发起和决议，更新系统状态
学习者（Learner）	跟随者（Follower）	Follower 用于接收客户请求并向客户端返回结果，在选主过程中参与投票
	观察者（ObServer）	ObServer 可以接收客户端连接，将写请求转发给 leader 节点。但 ObServer 不参加投票过程，只同步 leader 的状态。ObServer 的目的是为了扩展系统，提高读取速度
客户端（Client）		请求发起方

• Zookeeper的核心是原子广播，这个机制保证了各个Server之间的同步。实现这个机制的协议叫做Zab协

议。Zab协议有两种模式，它们分别是恢复模式（选主）和广播模式（同步）。当服务启动或者在领导者

崩溃后，Zab就进入了恢复模式，当领导者被选举出来，且大多数Server完成了和leader的状态同步以后

，恢复模式就结束了。状态同步保证了leader和Server具有相同的系统状态。

• 为了保证事务的顺序一致性，zookeeper采用了递增的事务id号（zxid）来标识事务。所有的提议（

proposal）都在被提出的时候加上了zxid。实现中zxid是一个64位的数字，它高32位是epoch用来标识

leader关系是否改变，每次一个leader被选出来，它都会有一个新的epoch，标识当前属于那个lea

der的
统治时期。低32位用于递增计数。

- 每个Server在工作过程中有三种状态：
  LOOKING：当前Server不知道leader是谁，正在搜寻
  LEADING：当前Server即为选举出来的leader
  FOLLOWING：leader已经选举出来，当前Server与之同步

其他文档：http://www.cnblogs.com/lpshou/archive/2013/06/14/3136738.html

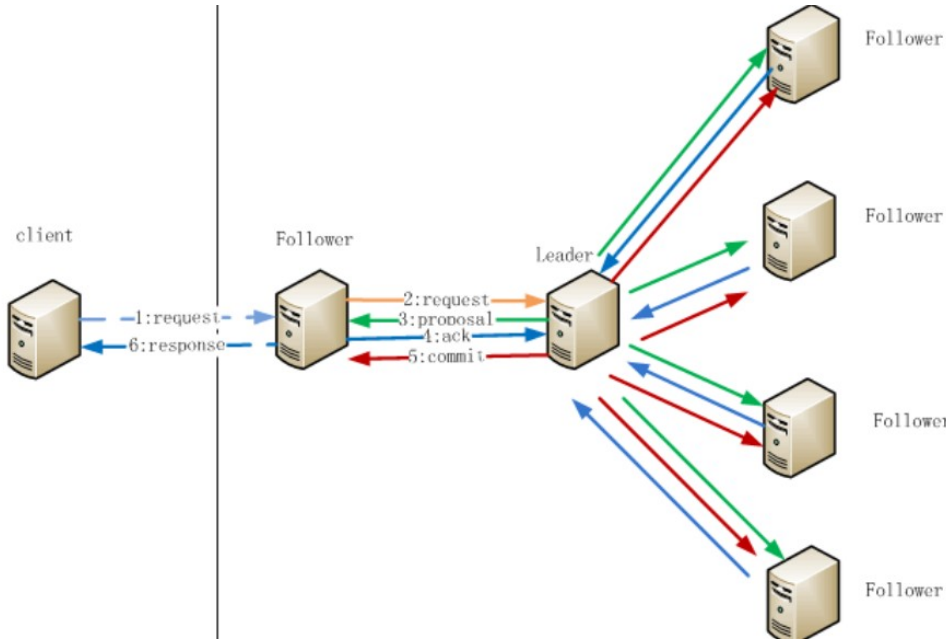
2、Zookeeper 的读写机制

» Zookeeper是一个由多个server组成的集群
» 一个leader，多个follower
» 每个server保存一份数据副本
» 全局数据一致
» 分布式读写
» 更新请求转发，由leader实施

3、Zookeeper 的保证

» 更新请求顺序进行，来自同一个client的更新请求按其发送顺序依次执行
» 数据更新原子性，一次数据更新要么成功，要么失败
» 全局唯一数据视图，client无论连接到哪个server，数据视图都是一致的
» 实时性，在一定事件范围内，client能读到最新数据

4、Zookeeper节点数据操作流程



注：1.在Client向Follwer发出一个写的请求

2.Follwer把请求发送给Leader

3.Leader接收到以后开始发起投票并通知Follwer进行投票

4.Follwer把投票结果发送给Leader

5.Leader将结果汇总后如果需要写入，则开始写入同时把写入操作通知给Leader，然后commit;

6.Follwer把请求结果返回给Client

- Follower主要有四个功能：
- 1. 向Leader发送请求（PING消息、REQUEST消息、ACK消息、REVALIDATE消息）；

图床好像挂了?

-----江北

2. Re:Zookeeper 3、Zookeeper工作原理（详细）
写的很好

--鹿慕叶

3. Re:Hadoop 2、配置HDFS HA (高可用)
很强，很好，大数据初学阶段，正好学习。非常感谢。

--枫若雪

4. Re:Java 判断字符串 中文是否为乱码
TransformUtils. 源码有没有，看看toString() 方法

--不想找对象

5. Re:Zookeeper 5、Zookeeper应用场景
客户端可以请求master 吗?

--CoobeeDior

阅读排行榜

1. Zookeeper 3、Zookeeper工作原理（详细）(30106)
2. CDH 1、CDH简介(26097)
3. Python的ASCII, GB2312, Unicode , UTF-8 相互转换(14639)
4. Impala 2、Impala Shell 和 Impala SQL(12853)
5. Java 获取Linux 的IP地址(11229)
6. HBase 1、HBase介绍和工作原理(10395)
7. Hive 2、Hive 的安装配置(本地MySql模式)(7640)
8. MySql 取一天的开始时间和结束时间(7605)
9. Eclipse Maven profiles 多环境配置，测试环境与开发环境分开打包(7477)
10. MyCat 主键ID自增长配置(7023)
11. HBase 4、Phoenix安装和Squirrel安装(6473)
12. Linux学习笔记1：配置Linux网络和克隆虚拟机并更改配置(6297)
13. Error: Linux下 mysql.sock文件丢失被删除解决方法(5809)
14. Hadoop 2、配置HDFS HA (高可用)(4861)
15. HBase 6、用Phoenix Java api操作HBase(4848)
16. Struts2 在Action中获取request、session、servletContext的三种方法(4750)
17. Hive 8、Hive2 beeline 和 Hive jdbc(4539)
18. Hadoop 3、Hadoop 分布式存储系统HDFS(4347)
19. Tengine笔记1：安装Tengine和Tengine说明(4279)
20. MyCat 安装部署，实现数据库分片存储(4053)

评论排行榜

1. Eclipse Maven profiles 多环境配置，测试环境与开发环境分开打包(3)
2. Zookeeper 5、Zookeeper应用场景(2)
3. Linux学习笔记1：配置Linux网络和克隆虚拟机并更改配置(2)
4. [转]Java汉字按照拼音排序(2)
5. Java 判断字符串 中文是否为乱码(1)
6. Zookeeper 3、Zookeeper工作原理（

- 2 .接收Leader消息并进行处理;
- 3 .接收Client的请求，如果为写请求，发送给Leader进行投票;
- 4 .返回Client结果。
- Follower的消息循环处理如下几种来自Leader的消息:
- 1 .PING消息： 心跳消息;
- 2 .PROPOSAL消息：Leader发起的提案，要求Follower投票;
- 3 .COMMIT消息：服务器端最新一次提案的信息;
- 4 .UPTODATE消息：表明同步完成;
- 5 .REVALIDATE消息：根据Leader的REVALIDATE结果，关闭待revalidate的session还是允许其接受消息;
- 6 .SYNC消息：返回SYNC结果到客户端，这个消息最初由客户端发起，用来强制得到最新的更新。

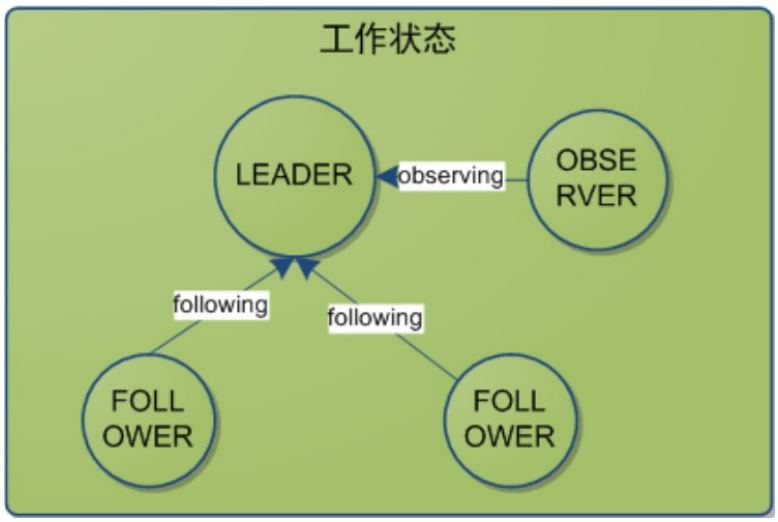
5、Zookeeper leader 选举

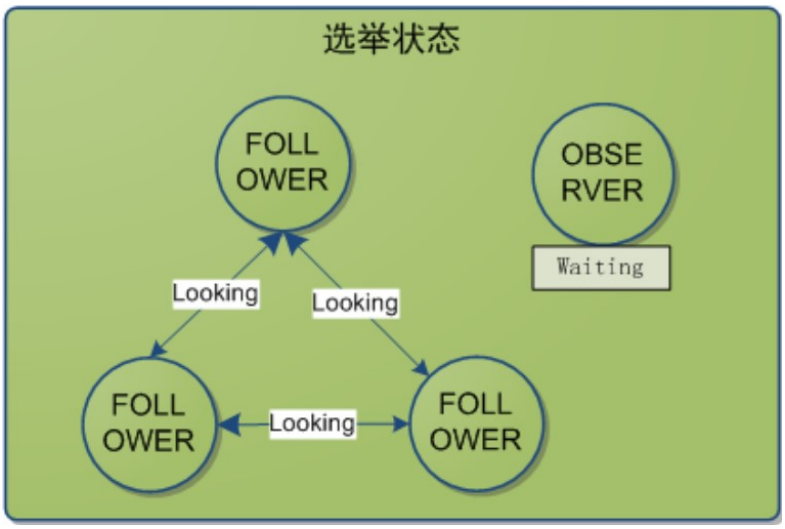
- 半数通过
  - 3台机器 挂一台 2>3/2
  - 4台机器 挂2台 2！>4/2

- A提案说，我要选自己，B你同意吗？C你同意吗？B说，我同意选A；C说，我同意选A。(注意，这里超过半数了，其实在现实世界选举已经成功了。

但是计算机世界是很严格，另外要理解算法，要继续模拟下去。)

- 接着B提案说，我要选自己，A你同意吗；A说，我已经超半数同意当选，你的提案无效；C说，A已经超半数同意当选，B提案无效。
- 接着C提案说，我要选自己，A你同意吗；A说，我已经超半数同意当选，你的提案无效；B说，A已经超半数同意当选，C的提案无效。
- 选举已经产生了Leader，后面的都是follower，只能服从Leader的命令。而且这里还有个小细节，就是其实谁先启动谁当头。





6、zxid

- znode节点的状态信息中包含czxid，那么什么是zxid呢?
- ZooKeeper状态的每一次改变，都对应着一个递增的Transaction id，该id称为zxid. 由于zxid的递增性质，如果zxid1小于zxid2，那么zxid1肯定先于zxid2发生.

创建任意节点，或者更新任意节点的数据，或者删除任意节点，都会导致Zookeeper状态发生改变，从而导致zxid的值增加.

7、Zookeeper工作原理

» Zookeeper的核心是原子广播，这个机制保证了各个server之间的同步。实现这个机制的协议叫做Zab协议。Zab协议有两种模式，它们分别是恢复模式和广播模式。

当服务启动或者在领导者崩溃后，Zab就进入了恢复模式，当领导者被选举出来，且大多数server的完成了和leader的状态同步以后，恢复模式就结束了。

状态同步保证了leader和server具有相同的系统状态

» 一旦leader已经和多数的follower进行了状态同步后，他就可以开始广播消息了，即进入广播状态。这时候当一个server加入zookeeper服务中，它会在恢复模式下启动，

发现leader，并和leader进行状态同步。待到同步结束，它也参与消息广播。Zookeeper服务一直维持在Broadcast状态，直到leader崩溃了或者leader失去了大部分

的followers支持。

» 广播模式需要保证proposal被按顺序处理，因此zk采用了递增的事务id号(zxid)来保证。所有的提议(proposal)都在被提出的时候加上了zxid。

实现中zxid是一个64为的数字，它高32位是epoch用来标识leader关系是否改变，每次一个leader被选出来，它都会有一个新的epoch。低32位是个递增计数。

» 当leader崩溃或者leader失去大多数的follower，这时候zk进入恢复模式，恢复模式需要重新选举出一个新的leader，让所有的server都恢复到一个正确的状态。

» 每个Server启动以后都询问其它的Server它要投票给谁。
» 对于其他server的询问，server每次根据自己的状态都回复自己推荐的leader的id和上一次处理事务的zxid（系统启动时每个server都会推荐自己）
» 收到所有Server回复以后，就计算出zxid最大的哪个Server，并将这个Server相关信息设置成下一次要投票的Server。
» 计算这过程中获得票数最多的的sever为获胜者，如果获胜者的票数超过半数，则改server被选为leader。否则，继续这个过程，直到leader被选举出来

» leader就会开始等待server连接
» Follower连接leader，将最大的zxid发送给leader
» Leader根据follower的zxid确定同步点
» 完成同步后通知follower 已经成为uptodate状态
» Follower收到uptodate消息后，又可以重新接受client的请求进行服务了

8、数据一致性与paxos 算法

• 据说Paxos算法的难理解与算法的知名度一样令人敬仰，所以我们先看如何保持数据的一致性，这里有个原则就是：

• 在一个分布式数据库系统中，如果各节点的初始状态一致，每个节点都执行相同的操作序列，那么他们最后能得到一个一致的状态。

• Paxos算法解决的什么问题呢，解决的就是保证每个节点执行相同的操作序列。好吧，这还不简单，master维护一个

全局写队列，所有写操作都必须 放入这个队列编号，那么无论我们写多少个节点，只要写操作是按编号来的，就能保证一

致性。没错，就是这样，可是如果master挂了呢。

• Paxos算法通过投票来对写操作进行全局编号，同一时刻，只有一个写操作被批准，同时并发的写操作要去争取选票，

只有获得过半数选票的写操作才会被 批准（所以永远只会有一个写操作得到批准），其他的写操作竞争失败只好再发起一

轮投票，就这样，在日复一日年复一年的投票中，所有写操作都被严格编号排 序。编号严格递增，当一个节点接受了一个

编号为100的写操作，之后又接受到编号为99的写操作（因为网络延迟等很多不可预见原因），它马上能意识到自己 数据

不一致了，自动停止对外服务并重启同步过程。任何一个节点挂掉都不会影响整个集群的数据一致性（总2n+1台，除非挂掉大于n台）。

总结

• Zookeeper 作为 Hadoop 项目中的一个子项目，是 Hadoop 集群管理的一个必不可少的模块，它主要用来控制集群中的数据，

如它管理 Hadoop 集群中的 NameNode，还有 Hbase 中 Master Election、Server 之间状态同步等。\

关于Paxos算法可以查看文章 Zookeeper全解析——Paxos作为灵魂

推荐书籍：《从Paxos到Zookeeper分布式一致性原理与实践》

9、Observer

• Zookeeper需保证高可用和强一致性；
• 为了支持更多的客户端，需要增加更多Server；
• Server增多，投票阶段延迟增大，影响性能；
• 权衡伸缩性和高吞吐率，引入Observer
• Observer不参与投票；
• Observers接受客户端的连接，并将写请求转发给leader节点；
• 加入更多Observer节点，提高伸缩性，同时不影响吞吐率

10、 为什么zookeeper集群的数目，一般为奇数个?

•Leader选举算法采用了Paxos协议；

•Paxos核心思想：当多数Server写成功，则任务数据写成功如果有3个Server，则两个写成功即可；如果有4或5个Server，则三个写成功即可。

•Server数目一般为奇数（3、5、7）如果有3个Server，则最多允许1个Server挂掉；如果有4个Server，则同样最多允许1个Server挂掉由此，

我们看出3台服务器和4台服务器的的容灾能力是一样的，所以为了节省服务器资源，一般我们采用奇数个数，作为服务器部署个数。

11、Zookeeper 的数据模型

» 层次化的目录结构，命名符合常规文件系统规范
» 每个节点在zookeeper中叫做znode,并且其有一个唯一的路径标识
» 节点Znode可以包含数据和子节点，但是EPHEMERAL类型的节点不能有子节点
» Znode中的数据可以有多个版本，比如某一个路径下存有多个数据版本，那么查询这个路径下的数据就需要带上版本
» 客户端应用可以在节点上设置监视器
» 节点不支持部分读写，而是一次性完整读写

12、Zookeeper 的节点

» Znode有两种类型，短暂的（ephemeral）和持久的（persistent）
» Znode的类型在创建时确定并且之后不能再修改
» 短暂znode的客户端会话结束时，zookeeper会将该短暂znode删除，短暂znode不可以有子节点
» 持久znode不依赖于客户端会话，只有当客户端明确要删除该持久znode时才会被删除
» Znode有四种形式的目录节点
» PERSISTENT（持久的）
» EPHEMERAL(暂时的)
» PERSISTENT_SEQUENTIAL（持久化顺序编号目录节点）
» EPHEMERAL_SEQUENTIAL（暂时化顺序编号目录节点）

分类: Zookeeper

Bodi
关注 - 5
粉丝 - 71
+加关注

4 0

« 上一篇: Zookeeper 2、Zookeeper的安装和配置（集群模式）
» 下一篇: Zookeeper 4、Zookeeper开发

posted @ 2016-03-16 22:49 Bodi 阅读(30107) 评论(1) 编辑 收藏

发表评论

#1楼 2018-07-04 13:37 | 鹿慕叶

写的很好

支持(0) 反对(

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。